Extracting Relations using SVO matching and BERT

Sai Koneru

May 20, 2020

1 Introduction

Books contain a lot of factual information and relations between several entities but they are in the form of unstructured text. In this project the corpus used is called "History of the World War: An Authentic Narrative of the World's Greatest War" (March and Beamish (1918)). It contains information about the world war 1 and describes relations between various countries and people around the world. Several legal and medical documents also contain several relationships between the entities that we want to highlight. For example, in medical domain we would like to know relationship between different genes but we cannot spend time in reading all the documents. We can use information retrieval and text mining techniques to analyze and structure the unstructured data.

Goal of the project is to build a community model that can show for instance, relationship between different who's and by traversing through these nodes we can also access the relevant passage where this relationship is mentioned. This will act as a tool to summarize the unstructured text and be effective in understanding higher level relationships. In Section 2 we will discuss the two methods used followed by their visualizations in Section 3. Finally we conclude our findings and present future work in Section 4.

2 Methodology

Relation Extraction is a task of extracting semantic relationships which occurs between different entities. There are several type of methods to extract relationships in text. In this project two approaches are used. A rule based approach is used to determine the subject,object and the verb phrase using dependency parsers. The second approach is to use BERT(Devlin, Chang, Lee, and Toutanova (2018)) that is trained in question answering to extract specific relation or possibly complex relations that can be formed by combining W's.

2.1 SVO Matching

After extracting these tuples we can build networks around these as predicates. We can constrain the subject and object on several type of entities and visualize the relationship among them. The architecture of the model is shown in Figure 1

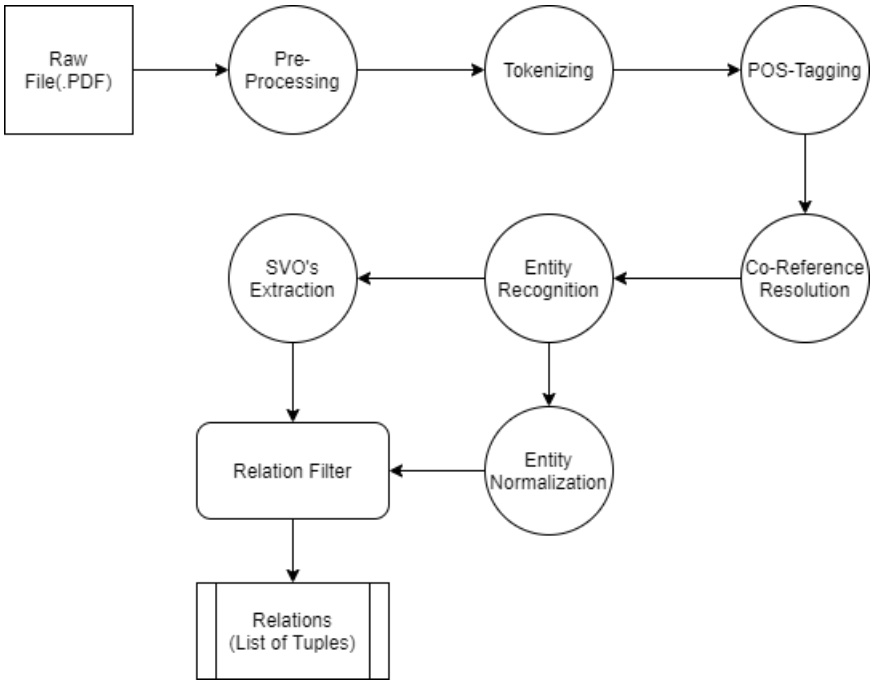

Figure 1: Data Flow of the Model.

For Part of speech tagging and entity recognition two implementations were tested. One implementation is using a conditional random field to detect the POS tags and entities and the other is to use the default model in spacy (Honnibal and Montani (2017)). We can also fine tune the pretrained model to increase precision and recall but as the corpus is a book that is written well unlike tweets, looking at the F1 score it is not needed unless we find a training corpus that matches the text that we are working on.

2.1.1 Pre-Processing and Tokenization

Corpus used in the project is available in PDF format. There are many issues with documents in this format such as, several tables that cannot be distinguished with normal text, words that end in between at the end of line which will then be treated as two words and meta data such as page numbers and other irrelevant text from the actual text in the book. We can filter the tables and page number by defining the patterns in which they occur and use regular expressions. However, we cannot know the last word in a sentence and the first word in the next sentence belong to the same word or not. In this implementation we considered them to be different words but this also causes some mistakes in the text.

For tokenizing the words, spacy (Honnibal and Montani (2017)) is used which does several tasks at once such as part of speech tagging, entity recognition and co-reference resolutions. These tasks are explained in detail in the further sections.

2.1.2 Part of Speech-Tagging

POS-tagging is the process of assigning a tag to every word that corresponds to the part of speech tag for that word in a sentence. It gives us a better representation to the words

when using them in a higher complexity tasks. One way it is done in the project is by using spacy.

Another way that was implemented is to use Conditional Random Fields. Conditional Random fields are statistical models that can be used as a classifier to predict labels. Entropy modelling is another method but it is not used as they suffer from labelling bias problem especially when we have less training data. We need to define features in order to predict the POS tags. The data that was used to train these tags is the treebank corpus[Marcus, Santorini, and Marcinkiewicz (1993)]. Several features were defined such as if all the letters in the words are capital, does it have any special prefix?, is it followed by a number?, is the word itself a number? etc. It can be seen in Table 4 that the average F1 score is 0.96. Any model that predicts with a F1 score of more than 0.8 we consider it to be satisfactory and enhance other elements in the pipeline.

2.1.3 Co-reference/Anaphora Resolution

Co-reference resolution is the task of resolving words in a text that refer to an entity. This is an important step if we want to capture all relations and have a good recall score. Several Linguistic techniques can be used to address this or deep learning which are the current state of the art method used right now. In this project neural coref(Lee, He, Lewis, and Zettlemoyer (2017)) is used which is in-built in spacy. This model works really well when it is given two sentences but performs poorly on large text as it is trained on small textual data. To mitigate this the text of the book is divided by paragraphs, these paragraphs are then resolved and joined together. The text from this will then be used for the following steps.

Some errors are also observed and few pronouns were not able to be resolved by the model. Temporal co-references are also not handled by this model and we need to use a rule based approach to fix these references. For example the sentence ”I woke up in the morning. Few hours after I went to the gym”. Here Few hours should be resolved to ”Few hours after morning”. Another example, the author of the corpus refers Germany as she in the text but the model couldn't resolve it. This is the case because it wasn't trained on this style of language.

2.1.4 Entity Recognition

Entity recognition is a task where want to tag the words that belong to a certain type. The number of types depend on a specific application but there are few general ones that are applicable everywhere. First we used spacy's in built model to recognize the entities that are applicable also in our context as we are working on a history book. It is trained on entities that we need such as countries,names etc.

A conditional random field model was also trained on a dataset in kaggle [1] which is a annotated corpus for named entity recognition in IOB format. Initially, several features

[1]https://www.kaggle.com/abhinavwalia95/how-to-loading-and-fitting-dataset-to-scikit/data

Table 1: Precision,Recall and F1 for NER

	Precision	Recall	F1
CRF	0.84	0.97	0.87
Spacy	0.86	0.96	0.90

were tried including the ones used in POS tagging but also the features incorporating the POS tags. After training this model it was observed that it was doing relatively well but had a problem with person names especially both in training and testing data set. Generally peoples name when referred in a text a suffix 's is added and also added a feature to see if the before and after words are capital because names are usually in capitals. The macro average scores for this model and spacy's Named Entity Recognition model is shown in Table 1. However even though it is above 0.8 the value does not seem to reflect the actual quality because most of the words are non entity's. As a result, by giving most of the words an 'O' tag it was able to get a high score as shown in Table 5. For further steps, spacy is used as it was better in all categories consistently rather than the CRF model and also has more classes of entities.

2.1.5 Entity Normalization

After detecting entities and organizing them by W's is not enough to use them in matching with the subjects and objects. We need to detect duplicate entities and resolve them into a single entity. This will make us have a clean database and compact representation when visualizing our knowledge graph. Different type of entities need to be resolved differently.

Dates can be resolved using standard tools that convert everything into one format. A dictionary is also used when it comes to resolving country names as they do not match in words but are just different names for the country. For example consider England and Great Britain, the similarity between these words is very low and can only be solved by dictionary. For the type of entities that can be resolved using similary measures several of them exist such as Levenshtein distance, Jaccard measure etc. However, the best method to normalize entities is to use SOFT-TFIDF as it takes into account the similarity of individual words and also is irrelevant to the order of the words. For this reason we have use this approach to normalize our entities.

2.1.6 SVO Extraction

Dependency parsing allows us to represent sentences grammatically and defines head word relationships which signifies how these words are modified. An example of dependency parser is shown in Figure 2.

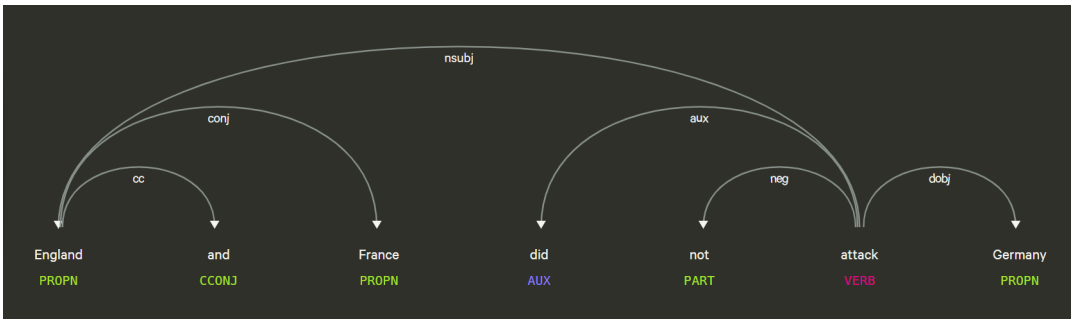

Figure 2: Example of Dependency parser with Negation and Conjunction

In Figure 2 we can see that we can extract the subject which is England , the verb is attack and the object is Germany. However, this does not give us accurate information. There are multiple issues with simply extracting the subject, verb and the object. The first issue is the boundary problem. Not only England, but France is also part of the subject and by creating rules that check the tree for conjunction, we can fix this. Another issue is the negation problem. The verb that we need to extract is !attack, so we have to check the verb for negations in the tree as well. However, this does not work always as there are multiple ways in text we can add negations and it is hard to create rules for every possible way. Sarcasm is also hard to detect but as this is a book this might not be a problem most of the time.

Now the idea is to extract a tuple of n-array from every sentence in the text. Some tuples may contain pronouns as subject or object. These might be because our co reference resolution model could not handle it or it is the author that is using pronouns for himself. We need to filter these kind of information as these do not help us understanding the information clearly but if we do keep it, we need to show the relevant passage to solve the ambiguity.

2.1.7 Relation Filtering

There are several complex relationships that we can try to analyze after collecting data in the form of subject, verb and objects. After we extracted all the relations from every sentence in the text, we can now filter these tuples to extract information that we want. For example, who-who relationship can be understood by combining different entities and using them as a filter on the relational data. We need to combine entities such as Persons, Geo-Political Entries into a single list and look for relations that are related to the who's. One way to do this is by checking if any word in the subject or object is present in the who's list. This is not a good approach as we can mention the same entity in different words. Similarly as we do in entity normalization, we compare each subject and object to our Geo-Political entities and then we check if the highest matching entity has a similarity score of more than 0.8 . If it is then we consider it a match and replace the subject/object with that corresponding entity.

Table 2: Precision, Recall and F1-scores for SVO-matching

	Precision	Recall	F1
SVO-Matching	0.84	0.35	0.48

2.1.8 Results and Limitations

Rule based approaches generally have low recall. This is the case because we cannot specify all different type of patterns that are possible. They also tend to have high precision because if it matches the pattern that we mentioned it is easy to extract the entities and their relation. Lets consider the sentence "Speaking to the Congress and the people of the United States, President Wilson made this declaration on November 11, 1918". This approach was able to extract tuples such as (Wilson,made,declaration),(Wilson,made,Nov 11,1918) but was unable to extract information such as (Wilson,spoke,Congress). This is not very good as we may loose valuable information and we are not aware of it as well. Another problem is that we cannot solve two predicates that mean the same but have different words. We need to use linguistic information to solve this. The scores for precision, recall and F1 evaluated on 20 sentences is shown in Table 2

2.2 BERT Question Answering for Extraction

Another method to extract relations is to use BERT which is fine-tuned for Question Answering. Question Answering(QA) is to give answer questions posed in natural language based on a collection of documents. Traditional methods focused on using linguistic techniques such as parsing, POS tagging and coreference resolution. Neural systems that use attention and LSTM/GRU layers were proposed with the advancements in deep learning. However, a recurrent neural network understands meaning of the word only looking at the contextual words in one direction. It only looks at the words that are before the center word or the words that follow it. In order to understand a true meaning of a word in a sentence, we need to define its meaning by looking at both directions. A bi-directional recurrent neural network gives a shallow representation of the sentence which is sub-optimal. Also, parallelism is not possible and the networks are relatively slow. Transformers (Vaswani et al. (2017)) are proposed later that solely use attention (luong2015effective) and feed forward networks which although have many parameters to train, parallel processing helps them to be faster. Several architectures based on transformers make a single model perform multiple NLP tasks including QA and only requiring addition fine-tuning to increase its performance in certain domain.

BERT can be trained to extract entities that have a certain relation. This means that we need to create a dataset either manually or make one from large databases. A new approach that we can consider is transfer learning where we use the BERT that is trained on QA dataset to extract any relation that we want. SQUAD (Rajpurkar, Zhang, Lopyrev, and Liang (2016)) is one of the popular datasets that was used for bench marking these models. One problem with this is that it does not have any questions that there is no answer to. It is

also important to know that the answer we are looking for is not in the context given. So, SQUAD 2.0 (Rajpurkar, Jia, and Liang (2018)) is a better dataset especially in our approach as we are using it to extract relations if there exists one. The architecture can be seen in Figure 3.

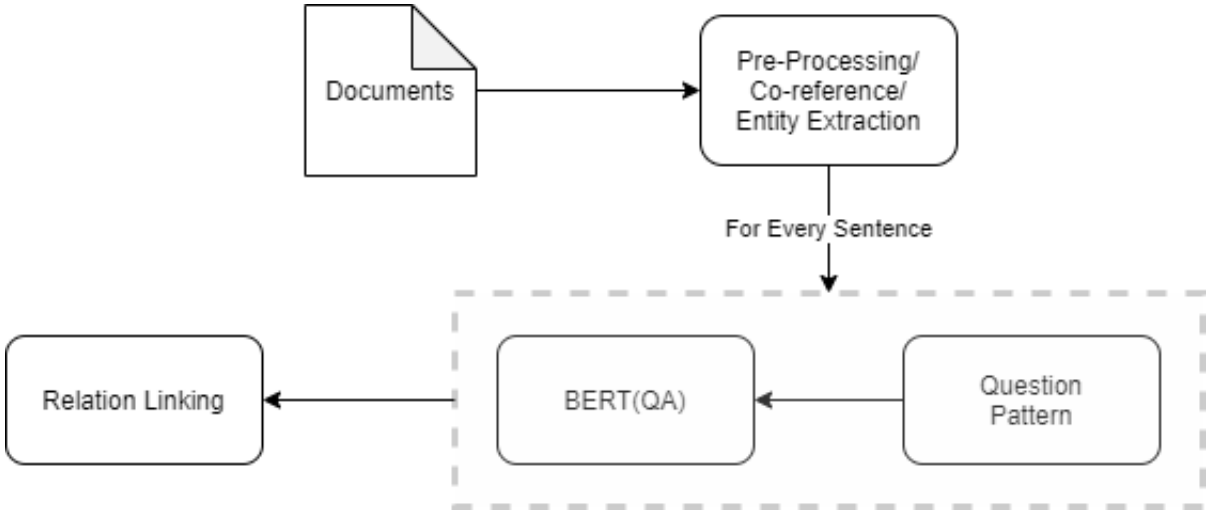

Figure 3: Data Flow for extraction using BERT

The initial steps are similar to the approach used before. We perform several pre-processing tasks, tokenization, co-reference resolution and entity extraction. After this lets say we want to find a relation attacked, that is to see which countries attacked. The steps that the model goes through to extract this is

- Recognize which entities are countries.
- Define a question pattern such as who attacked ENTITY?
- On all the sentences with countries, replace ENTITY with the country detected and use it as a question for BERT while giving the sentence as context.
- If BERT is able to answer, then link the answer with relation as attacked.

After we do these steps, we also use SOFT-TFIDF to normalize the entities and eliminate duplicates. Here we can see that it is crucial to use a model that was fine-tuned on SQUAD 2.0 as most of the sentences may not have an answer.

2.2.1 Results and Limitations

There are several issues with taking this approach. One major problem is that BERT predicts the answer in the context by predicting where the answer starts and ends. It cannot give a word in the beginning and another in the end. For this reason we are limited to define a question patterns that looks for the answer directly. If we ask a complex question like what happened in 1910, we cannot guarantee that we get all the relations and recall will be low. The architecture must me modified in order to handle this. Another issue with

Table 3: Precision, Recall and F1-scores for BERT-QA finding who attacked who

	Precision	Recall	F1
BERT-QA	0.75	0.89	0.81

using BERT in general is negations. It cannot handle them as the vector space does not really differentiate the words and their negated forms. For example 'really' and 'not really' occur in the same context more often than not. We used dependency parses to remove the sentence if it has negation. It seems to also get confused when a country attacks another country in a different country. For example if USA attacks Germany in France, BERT seems to get confused and the answer to the question who attacked France is USA. Maybe this can be resolved by asking question such as which country did ENTITY attack?. Although with these limitations, we were able to detect "attacked" relation without having to create a dataset and train from scratch. Evaluation of these approach for a small annotated sample is shwon in Table 3. These approach can be use-full when we want to find entities that are related but we do not have any dataset to train conventional BERT architectures that are used for relation extraction.

3 Visualization

3.1 WHO-WHO

This approach can now be used to visualize who were connected to who. SO, we combine different entities such as PERSONS, GEO-POLITICAL ENTITIES into one type of category. Now we can filter our relations based on the condition that the subject and object are of WHO type. This visualization can be seen in Figure 4.

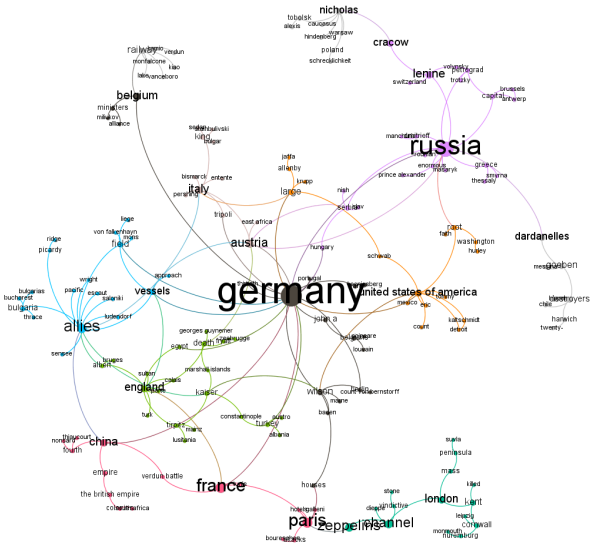

Figure 4: WHO-WHO

We can see that Germany is in the middle of our graph. This makes sense as they were the most important country and the book would be focusing more on germany compared to other countries. We can also see other big nodes such as Russia, Allies, France and England. This also seems valid as they are the major countries that were on the opposite side of Germany. Now if we zoom in on the graph we can also obtain more information. For example, if we were interested in Nicholas from Russia we can also get more information about him with this simple approach. In Figure 5 we can see the edges as predicates and all people/countries he is connected with. This gives us a higher level view of the corpus and can be used as a tool to find connections between different people in a large corpora.

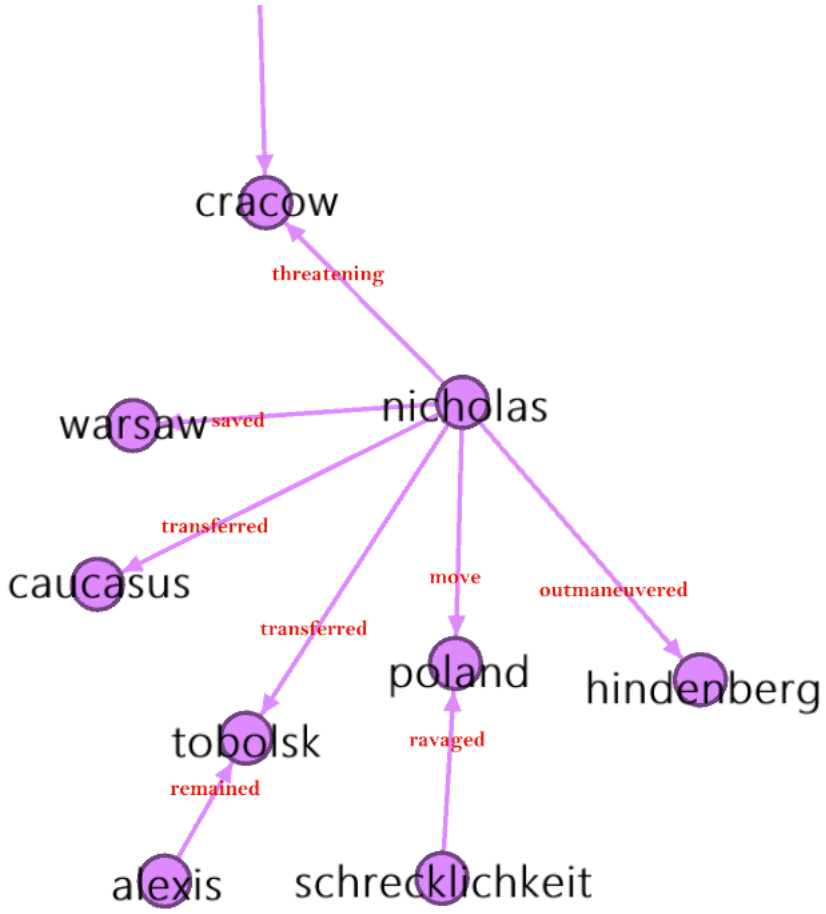

Figure 5: Tuples formed with Nicholas

3.2 WHO ATTACKED WHO

In order to find out which countries attacked which other countries we can use the approach proposed with BERT. we define the question pattern as who attacked entity and run the algorithm on the corpora. The result can be seen in Figure 6. Here a circle layout is used to show a compact representation. The layout used allows us to enter number of

nodes that should be displayed inside the circle and which then can be considered top important nodes. We can also see that the countries which played a major role are inside the circle.

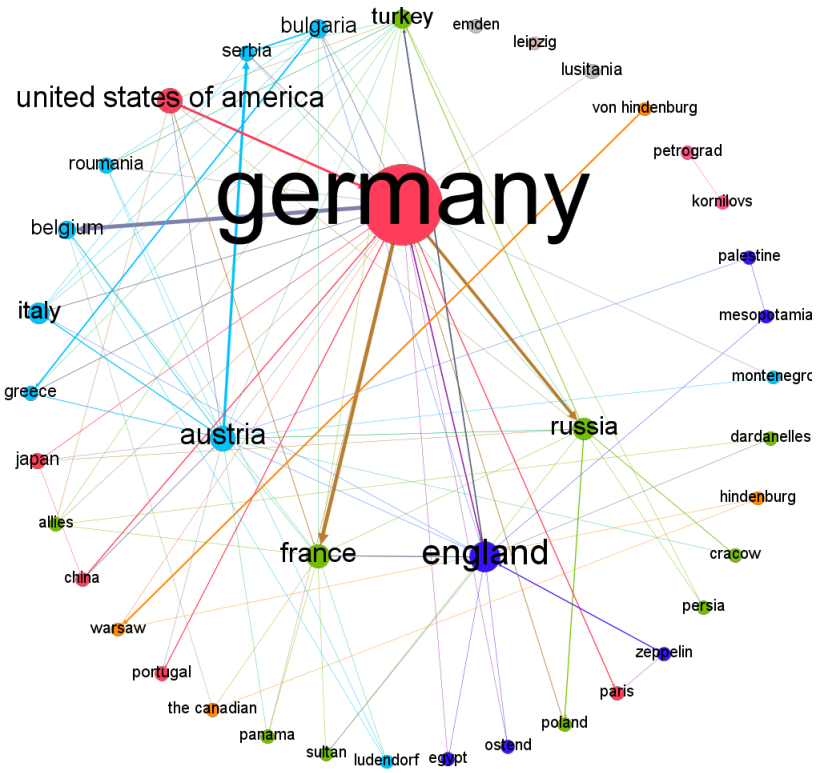

Figure 6: WHO ATTACKED WHO

One interesting fact about who attacked who is that one country can attack another country multiple times. This is represented in the graph by the how strong the color of an edge is. So in order to get more insight we applied few filters on this graph. They are to show which edges have more than a weight of one and also eliminate nodes that have a degree of 1 as they eliminate the least important nodes. The results can be seen in Figure 7. One interesting thing that needs to be analyzed in more detail is that there is an edge between England and France. The sentence that was used to form this relation from the corpora is "By this time Britain had thoroughly learned her lesson, and now countless shells and guns were pouring into France from Great Britain where thousands of factories, new and old, toiled night and day." BERT considers this a sentence where England attacks France even though it is not the case. But, it seems reasonable that given the context and with no other knowledge one might make this relation.

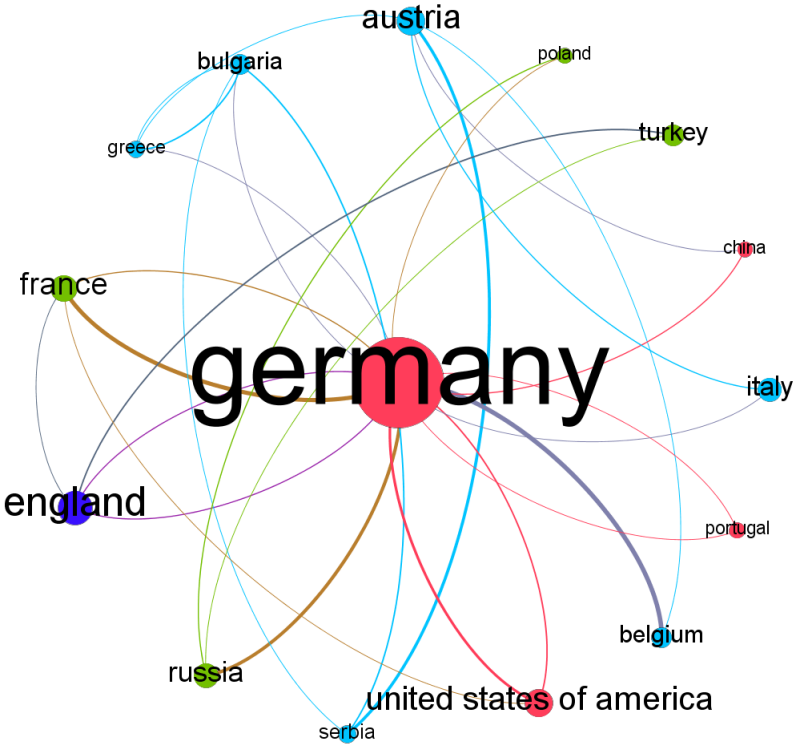

Figure 7: Countries who attacked each other multiple times

4 Conclusion

Two approaches were shown with one using rules and another with Deep Learning. The rules based approach has a very low recall and things we could do to improve it is to add more rules to detect the subject and object. Also the predicates may mean the same but are different words which needs to be further resolved. It still is an advantage as we extract all the information and can filter out as needed. Approach with BERT can only work if the question is straight-forward. The more the question is abstract like who did what?, the lower the chances of the answer consisting of entity. However, it still seems to work well without any domain specific data as it was trained on Wikipedia articles when asking precise questions. For Future work, it is interesting to see how we add/modify a layer in BERT so we can train it to predict answers from two different locations in the context. This may overcome the difficulties in answering complex questions.

References

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Honnibal, M., & Montani, I. (2017). spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. *To appear*.

Lee, K., He, L., Lewis, M., & Zettlemoyer, L. (2017). End-to-end neural coreference resolution. *arXiv preprint arXiv:1707.07045*.

March, F. A., & Beamish, R. J. (1918). *History of the world war: An authentic narrative of the world's greatest war*. John C. Winston Company.

Marcus, M., Santorini, B., & Marcinkiewicz, M. A. (1993). Building a large annotated corpus of english: The penn treebank.

Rajpurkar, P., Jia, R., & Liang, P. (2018). Know what you don't know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*.

Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2016). Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998–6008).

A POS Tagging

Table 4: Pos Model Precision, Recall and F1 per Class

	f1-score	precision	recall	support
#	1.000000	1.000000	1.000000	2.000000
$	1.000000	1.000000	1.000000	242.000000
”	1.000000	1.000000	1.000000	78.000000
,	1.000000	1.000000	1.000000	930.000000
-LRB-	1.000000	1.000000	1.000000	26.000000
-NONE-	0.999627	0.999254	1.000000	1340.000000
-RRB-	1.000000	1.000000	1.000000	26.000000
.	1.000000	1.000000	1.000000	762.000000
:	1.000000	1.000000	1.000000	77.000000
CC	0.997669	0.997669	0.997669	429.000000
CD	0.998546	0.999030	0.998062	1032.000000
DT	0.994101	0.994410	0.993793	1611.000000
EX	0.933333	0.875000	1.000000	7.000000
IN	0.979353	0.974632	0.984119	1952.000000
JJ	0.885202	0.863517	0.908004	1087.000000
JJR	0.846154	0.825000	0.868421	76.000000
JJS	0.891566	0.822222	0.973684	38.000000
MD	0.995098	0.995098	0.995098	204.000000
NN	0.950096	0.958596	0.941744	2901.000000
NNP	0.954495	0.953437	0.955556	1800.000000
NNPS	0.436782	0.826087	0.296875	64.000000
NNS	0.963731	0.938604	0.990240	1127.000000
PDT	0.666667	1.000000	0.500000	6.000000
POS	0.997455	0.994924	1.000000	196.000000
PRP	1.000000	1.000000	1.000000	222.000000
PRP$	1.000000	1.000000	1.000000	122.000000
RB	0.904051	0.921739	0.887029	478.000000
RBR	0.384615	0.500000	0.312500	16.000000
RBS	0.750000	1.000000	0.600000	5.000000
RP	0.722222	0.684211	0.764706	34.000000
TO	1.000000	1.000000	1.000000	464.000000
VB	0.952187	0.964948	0.939759	498.000000
VBD	0.952894	0.952894	0.952894	743.000000
VBG	0.932563	0.937984	0.927203	261.000000
VBN	0.908277	0.918552	0.898230	452.000000
VBP	0.890244	0.895706	0.884848	165.000000
VBZ	0.954617	0.968254	0.941358	324.000000
WDT	0.990476	0.981132	1.000000	104.000000
WP	1.000000	1.000000	1.000000	26.000000
WP$	1.000000	1.000000	1.000000	4.000000
WRB	0.981132	1.000000	0.962963	27.000000
“	1.000000	1.000000	1.000000	81.000000
accuracy	0.964669	0.964669	0.964669	0.964669
macro avg	0.924123	0.946260	0.916066	20039.000000
weighted avg	0.964092	0.964663	0.964669	20039.000000

B Named Entity Recognition

Table 5: NER Model Precision, Recall and F1 per Class for annotated chapter

	f1-score	precision	recall	support
B-art	0.666667	0.500000	1.000000	1.000000
B-geo	0.878049	0.857143	0.900000	20.000000
B-gpe	1.000000	1.000000	1.000000	16.000000
B-org	0.857143	0.750000	1.000000	9.000000
B-per	0.571429	0.666667	0.500000	4.000000
B-tim	1.000000	1.000000	1.000000	5.000000
I-art	1.000000	1.000000	1.000000	9.000000
I-geo	1.000000	1.000000	1.000000	9.000000
I-org	0.800000	0.666667	1.000000	4.000000
I-per	0.800000	0.666667	1.000000	2.000000
I-tim	0.923077	1.000000	0.857143	7.000000
O	0.993606	0.997433	0.989809	785.000000
accuracy	0.985075	0.985075	0.985075	0.985075
macro avg	0.874164	0.842048	0.937246	871.000000
weighted avg	0.985614	0.987421	0.985075	871.000000